

Serial Number 10/070,444

REMARKS

Reconsideration of the application is respectfully requested for the following reasons:

1. Formalities

The specification has been revised to be in proper U.S. format, including the addition of headers. Because the changes are all formal in nature, it is respectfully submitted that they do not involve new matter.

2. Rejection of Claim 4 Under 35 USC §112, 2nd Paragraph

This rejection has been addressed by amending claim 4 to eliminate the plural recitation return addresses for the calling function, as suggested by the Examiner.

3. Rejection of Claims 1 and 4 Under 35 USC §102(b) in view of "Defensive Programming in Rapid Development of a Parallel Scientific Program" (Cheng)

This rejection is respectfully traversed on the grounds that the Cheng publication does not disclose or suggest:

- a. a calling program forming a first checksum when calling a subprogram, the called program (*i.e.*, the subprogram) forming a second checksum and comparing it with the first checksum before execution of the called program in order to check for disturbances or tampering, as recited in claim 1; or
- b. a called program verifying a return address and stopping itself if the return address is not present in a table, as recited in claim 4.

Instead, the Cheng publication merely discloses the use of checksums to compress arrays without losing meaningful debugging information, and not the generation of checksums by a *called* program of the checksums to verify its own integrity. In particular, the Cheng patent fails to disclose any sort of self-verification and termination of a subprogram based on a checksum generated from data passed by the calling program before execution of the subprogram.

Serial Number 10/070,444

Line 6 of Fig. 5 of the Cheng publication calls for simulating "my set of particles" *before* calculating checksums and after calling the multiple parallel processes. The second paragraph of page 666 explains that this instruction performs "a portion of the problem that an individual process works on." Therefore, the Cheng publication clearly does not generate a checksum based on data passed by the calling program to the called program *before execution of the called program*, as claimed.

This difference is not merely trivial or coincidental, but relates to a fundamental difference between the claimed checksums and the so-called "checksums" of Cheng. Instead, of using checksums to look for corrupted data based on changes in the data, the Cheng checksums are based on *invariable physical constants* (see page 667, paragraph 3 of the Cheng publication), and are used as a way of compressing data. As explained in the left column, fifth full paragraph, on page 667 of the Cheng publication, "Computing a checksum [...] is a way to compress the information into a single value." The checksum is not used, in this instance, to check for changes in what Cheng refers to as *dynamic variables*, but rather to store *static variables* during the compression process, and therefore cannot be used to verify passage of the variables between successive programs, but only to compare the variables between parallel programs.

It is true that this involves passing variables from one program to another. However, the checksums are used to verify duplication of constants between *parallel programs* to which the variables are passed, and not passing of variables from a calling program to a called program. As explained in line 4 of paragraph 1 on page 668 of the Cheng patent, the master process forks (starts) *n* slave processes after a checksum of the *static variables* is calculated in line 2.1 of the program listing of Fig. 5 and, subsequently in line 6.1, each of the slave processes again calculates a checksum for the same static variables and in line 6.2 checks their invariance with respect to the checksums calculated previously by the master. On the other hand, the checksums calculated in line 9.1 of Fig. 5 from the dynamic variables are logged into a special file and used to verify function for the parallel program. As explained in the first paragraph on page 668, the

Serial Number 10/070,444

values of these checksums will be independent of the number of parallel slave processes if the parallel program is functioning properly. The checksums illustrated in Fig. 5 of Cheng are thus either used to verify simulation of physical processes by checking invariant constants or, in effect, used as a reference for testing different execution paths by comparing the variable between the parallel programs.

Basically, checksums are normally used to check for corrupted data transferred via an insecure channel, effectively representing lossy codings, and therefore cannot be used for loss-less compression. The checksums of Cheng, on the other hand, are used for loss-less compression. See, *e.g.*, the abstract of the Cheng publication. Because the checksums of Cheng do not correspond to those of the claimed invention, and in particular because the checksums of Cheng are not formed and compared before execution of the called program, as recited in claim 1, it is respectfully submitted that the Cheng publication does not anticipate claim 1.

In addition, with respect to claim 4, it is noted that the Examiner indicates on page 6 of the Official Action, in the section entitled "Allowable Matter," that if claim 4 specified that the return address is the parameter for which the checksum is created, then claim 4 would be allowable. While it is agreed that the return address verification-by-table feature recited in claim 4 is allowable and not disclosed in the Cheng patent, it is respectfully submitted that the Examiner has misinterpreted the return address of this embodiment as the parameter for which the checksum is created. The return address is in fact not the parameter for which the checksum is created.

Instead, the specification describes verification of the return address as being supplemental or alternative to verification of the checksum. Instead, of comparing a checksum generated by the calling program with one generated by the called program, this embodiment has the called program simply look for the return address in a table and cease execution if the return address is not present. Unlike the checksum embodiment, in which the called program *generates* the second checksum, the return address is not generated by the called program. Therefore, the

Serial Number 10/070,444

return address verified by the called program (and received from the calling program) as recited in claim 1 is not the same as either of the checksums of claim 1, and of course does not correspond to any of the "checksums" taught by the Cheng publication.

In summary, the parallel checksum verification of Cheng is not equivalent to the claimed check of modularly constructed sequential programs (*i.e.*, programs and subprograms), and the checksums of Cheng are not analogous to those of the claimed invention. As a result, the difference between the checksum verification described in the Cheng publication and the verification recited in claims 1 and 4, namely that verification occurs before execution of the called program, is significant and the Cheng patent neither anticipates nor suggests the claimed invention. Withdrawal of the rejection of claims 1 and 4 in view of the Cheng patent and an indication of the allowability of claims 1 and 4 is therefore respectfully requested.

4. Rejection of Claims 2 and 5 Under 35 USC §102(b) in view of U.S. Patent No. 5,761,414 (Akaishi)

This rejection is respectfully traversed on the grounds that the Akaishi patent does not disclose or suggest timing execution of a subprogram and shutting it down if execution exceeds a predetermined number of cycles, as recited in claims 2 and 5, by:

- a. counting cycles (the timer circuit of Akaishi checks an elapsed time T_d rather than a number of cycles); and
- b. comparing the number of cycles with a preset number (the reference "value" T_c is incremented only upon execution of an interrupt routine, rather than being a fixed number that can be compared with the elapsed time or number of cycles).

Instead of the claimed number of cycles, Akaishi refers to "time periods" (see col. 3, and col. 2, line 46. This strategy only works if the program to be checked is the only process being executed on the processor. If the processor is running more than one program, then an absolute timer will take into account the other running processes and not reflect the actual execution of the subroutine being checked. On the other hand, since the claimed invention uses the number of

Serial Number 10/070,444

cycles rather than an absolute count, the claimed invention will accurately catch any variations in the program that affect execution time.

Secondly, as explained in the paragraph cited by the Examiner (col. 3, lines 16-52), the reference count T_c is implemented by an integer counter which is incremented upon each execution of an interrupt routine. This number is highly variable and will obviously depend on the frequency of calls of the interrupt routine. Like the absolute time T_d , it is not useful for verifying execution of a subprogram.

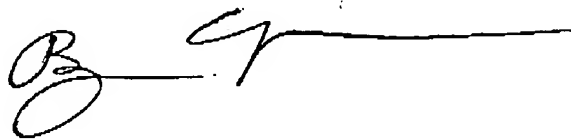
Because the Akaishi patent does not disclose or suggest verification of a subprogram run based on number of execution cycles, as claimed, it is respectfully submitted that the Akaishi patent does not anticipate claims 2 and 5, and withdrawal of the rejection of claims 2 and 5 under 35 USC §102(b) is respectfully requested.

5. Rejection of Claim 3 Under 35 USC §103(a) in view of "Defensive Programming in Rapid Development of a Parallel Scientific Program" (Cheng)

This rejection is respectfully traversed on the grounds that the Cheng publication fails to disclose or suggest, for the reasons described above, the limitations of claim 1 from which claim 3 depends.

Having thus overcome each of the rejections made in the Official Action, withdrawal of the rejections and expedited passage of the application to issue is requested.

Respectfully submitted,
BACON & THOMAS, PLLC



By: BENJAMIN E. URCIA
Registration No. 33,805

Date: December 22, 2004

Serial Number 10/070,444

BACON & THOMAS, PLLC
625 Slaters Lane, 4th Floor
Alexandria, Virginia 22314

Telephone: (703) 683-0500

NWD 23706089 Pending A.../EZUblablwz ScrlOTD44'adl wpa